Hands-On

Performance and Graphics Tuning Your Java Application

Session 632

These are confidential sessions—please refrain from filming or taking pictures.

Enterprise IT

Hands-On
Performance and Graphics Tuning Your Java Application

Viktor Miladinov
Java Classes Engineer

http://developer.apple.com/wwdc2005/

What's the Purpose of this Session?

- Make your Java Apps run faster on Mac OS X
- How do you do that?
- Graphics performance tips
- Mac OS X tools in Tiger that you can use to profile your App

Outline

- Graphics do’s and don’ts
- QuartzDebug—repaints profiling
- Shark for Java—profiling tool

Graphics Do’s and Don’ts

Gerard Ziemski
Java Classes Engineer

http://developer.apple.com/wwdc2005/

Java2D on Mac OS X Overview

- Java2D on Mac OS X uses Quartz
- Some Java2D operations don’t map directly to Quartz
  - Slow path
- Choose APIs to avoid the slower paths and hit the fast paths
- Cross platform graphics code may have different performance characteristics when run on Mac OS X

Outline (Avoid Hitting the Slow Path!)

- Buffered images tips
- On-screen painting and multiple threads
- XOR vs. alpha compositing

BufferedImages

- Call `createCompatibleImage() !`
- Quartz2D doesn’t let us look at pixels directly (immutable images)
- Avoid calling `getRGB or Raster.getDataBuffer` when not absolutely necessary:
  - `getRGB` is slow on Mac OS X
  - `Raster.getDataBuffer` will cause all subsequent drawing to use the slow path

Hands-On
Zombies

Scott Adler

http://developer.apple.com/wwdc2005/

Painting from the Wrong Thread

- All windows are double buffered on Mac OS X:
  - We need to flush the back buffer at some point, but when?
- Painting on-screen from the wrong thread can cause flicker
- Painting an image at the wrong time can cause flicker too

Hands-On Pong

Gerard Ziemski

http://developer.apple.com/wwdc2005/

Painting from the Wrong Thread

- Other solutions
  - `BufferStrategy`

XOR Hits the Slow Path

- XOR is not natively supported by Quartz
  - Custom implementation of XOR
  - Slower than other Quartz operations
  - May cause your app to loose hardware acceleration
  - Resulting XOR colors are not what you might expect
- Use alpha compositing instead
  - More flexible
  - Faster
  - Compatible with graphics frameworks

XOR Hits the Slow Path

Instead of:

```
void drawHighlight(Graphics2D g2, int x, int y, int w, int h)
{
    g2.setColor(Color.blue);
    g2.setXORMode(Color.black);
    g2.fillRect(x, y, w, h);
}
```

Use:

```
AlphaComposite ac = AlphaComposite.getInstance
(AlphaComposite.SRC_OVER, 0.5f);
void drawHighlight(Graphics2D g2, int x, int y, int w, int h)
{
    g2.setColor(Color.blue);
    g2.setComposite(ac);
    g2.fillRect(x, y, w, h);
}
```

QuartzDebug

Viktor Miladinov

http://developer.apple.com/wwdc2005/

Animating a Rectangle

- How to animate a rectangle

Animating a Rectangle

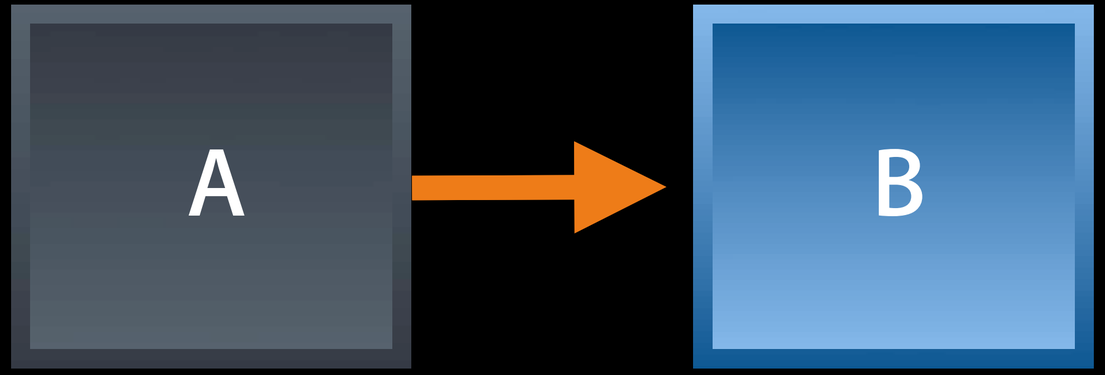
- Assume we subclass Component
- We override public void paint (Graphics g)

Animating a Rectangle

Step 1: Change the location of the rectangle from A to B

Step 2: Ask Java's Repaint Manager to repaint us

Step 3: On paint() draw the rectangle to the screen



What Could Go Wrong?

- Are you repainting more often then you need to?
- Are you repainting a larger area than you need to?
- QuartzDebug has the answers
- /Developer/Applications/Performance Tools

Hands-On QuartzDebug

Viktor Miladinov

http://developer.apple.com/wwdc2005/

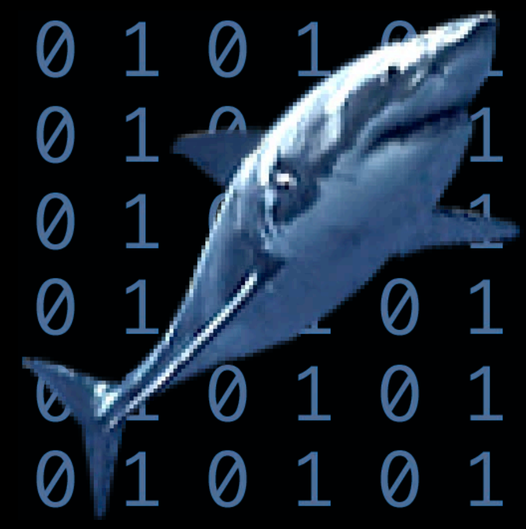
Shark for Java

High Level Performance Analysis

Christy Warren
Tiger Responsiveness DRI

http://developer.apple.com/wwdc2005/

What Is Shark?



- Profiling tool
- Profile a running process, thread or entire system
- Java Profiling Options—process only
  - Timed Samples—find hot spots
  - Allocations—view memory usage
  - Methods—exact trace of method calls
- Non Java Profiling
  - Time, memory, function, hardware/system event
  - Useful to study JNI calls

Shark—Key Features

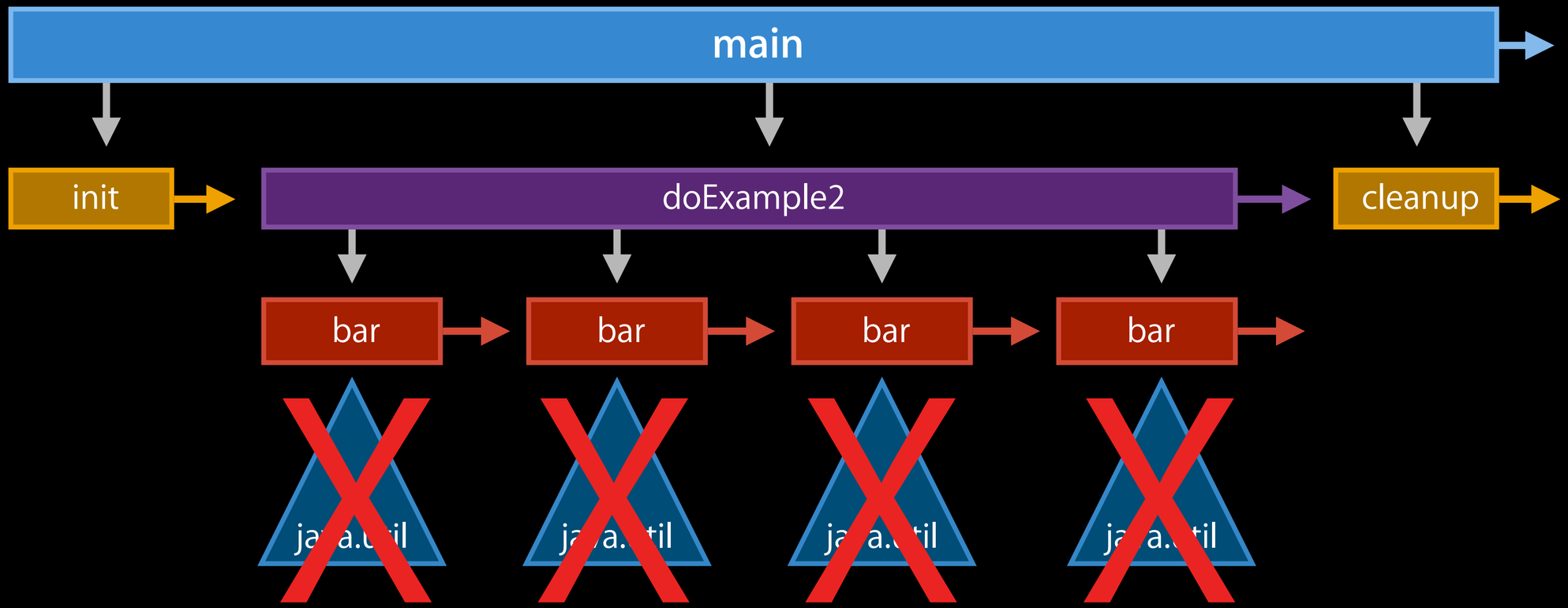
- Profile View
  - Find Hot Spots
  - Simultaneous “Heavy” and “Tree” perspectives
- Chart View
  - Visualize execution
- Code Browser
- Remote profiling over a network

Shark—Key Features

- Data Mining
  - Eliminate what you don’t want
    - Charge Package to Caller
    - Charge Symbol to Caller
  - See what you do want
    - Focus Symbol
    - Focus Package

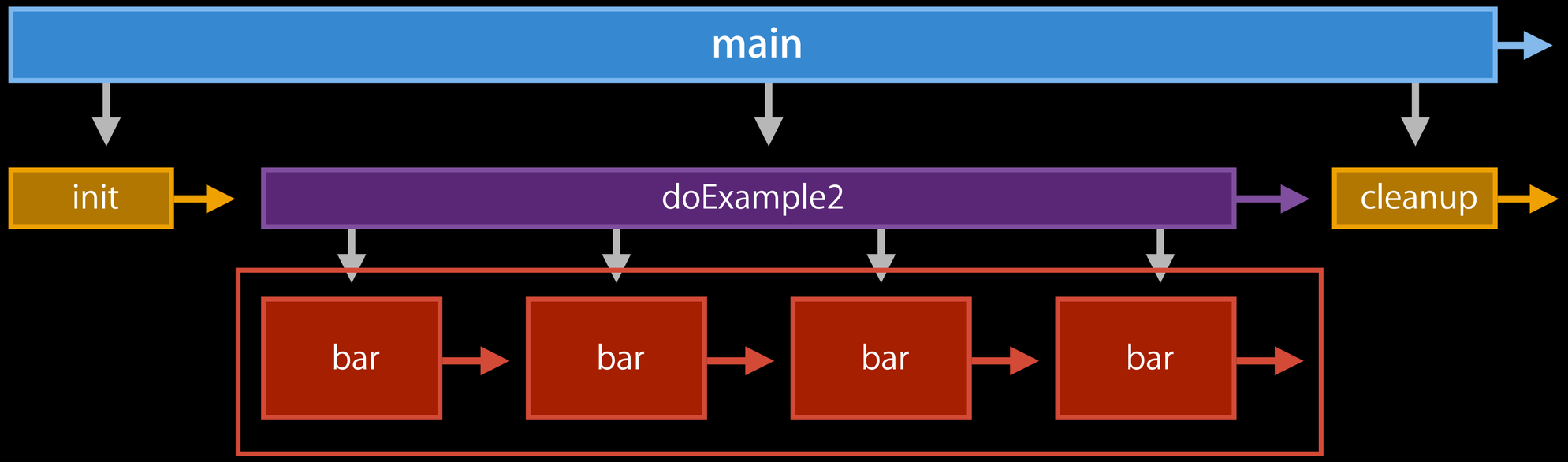
Charging a Package To Caller

Shark calls them Libraries



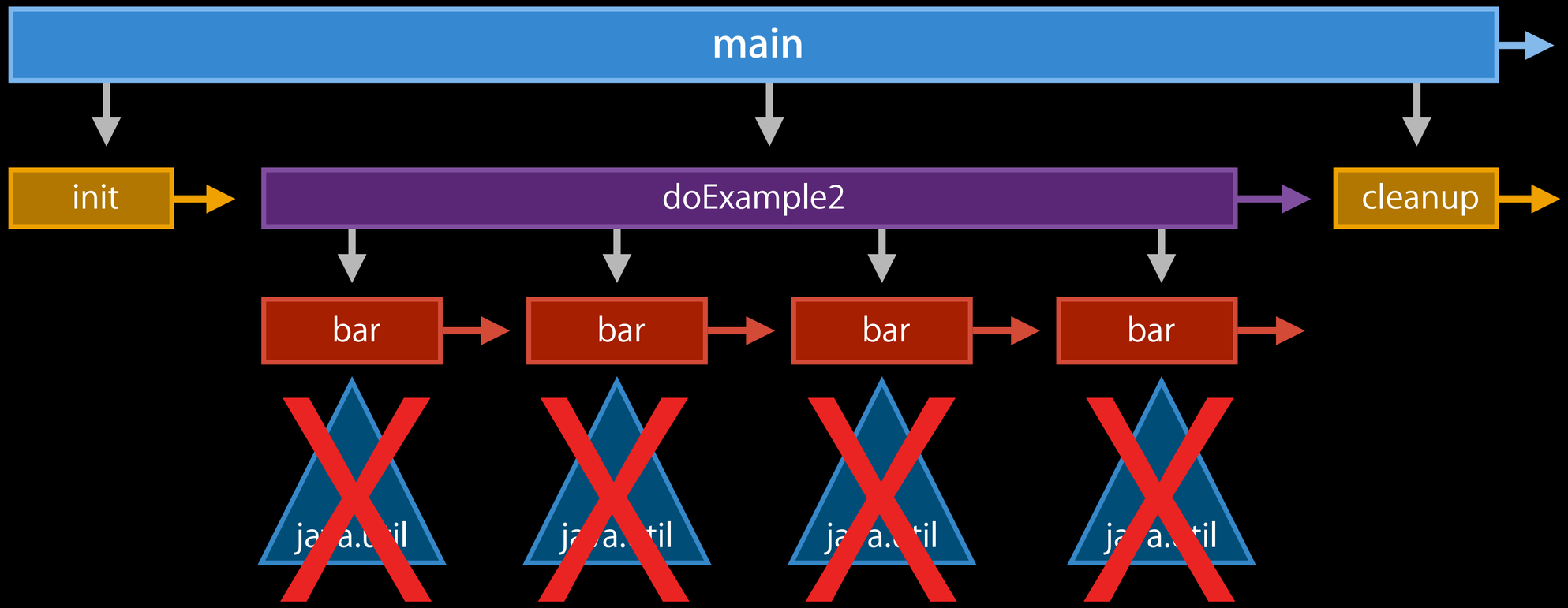
Charging a Package To Caller

Excluding a Package



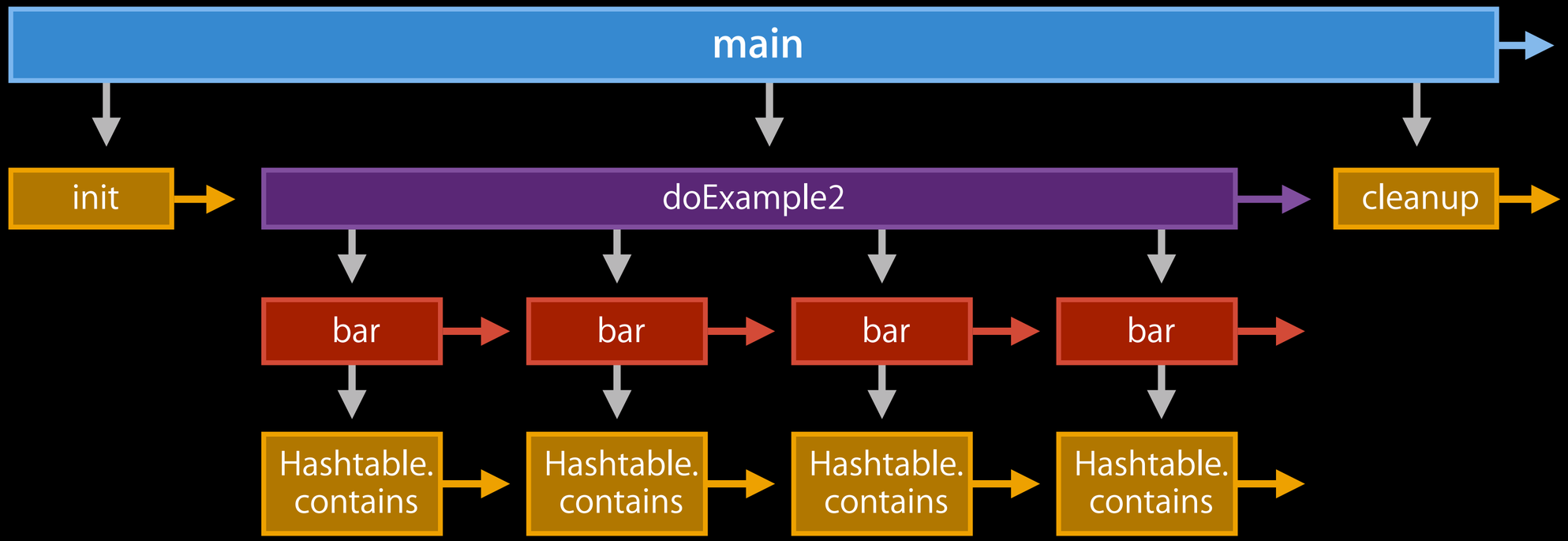
Flatten Package to Entry Point

Flattening a Package



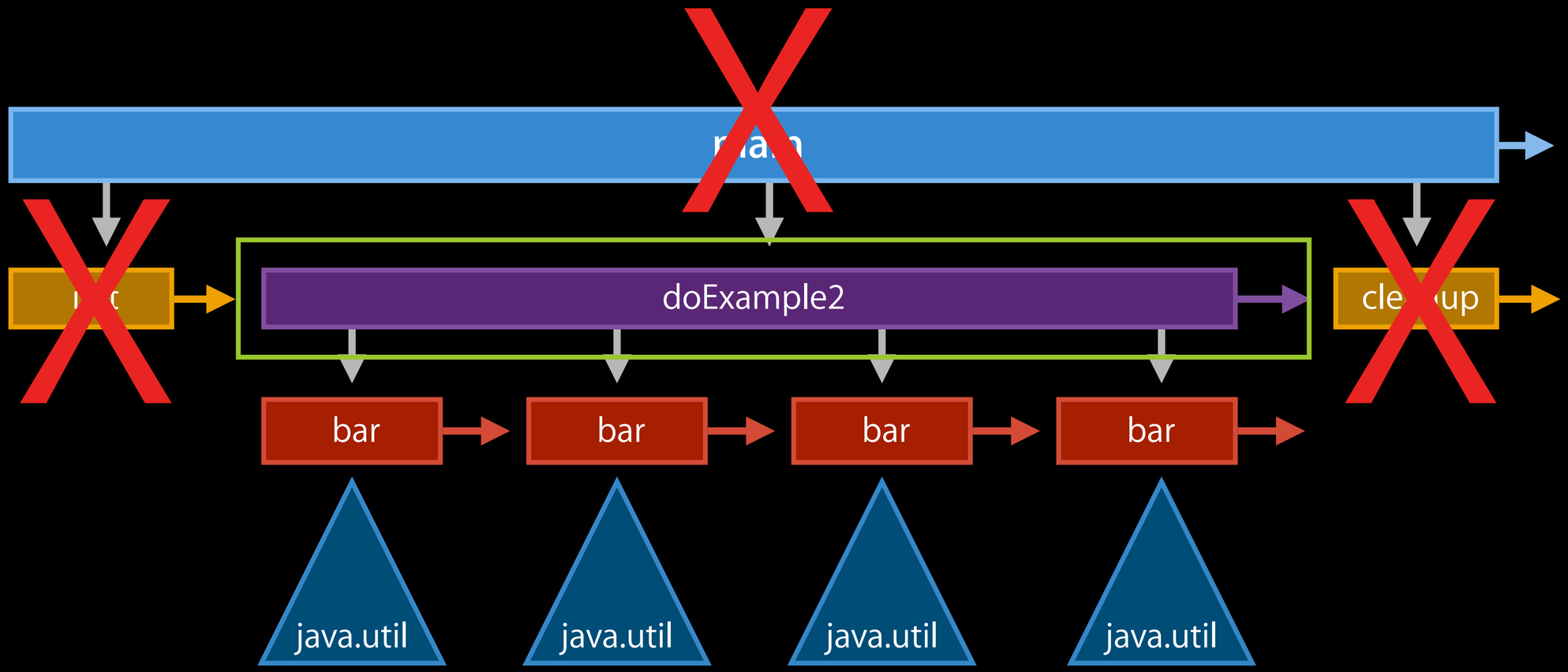
Flatten Package to Entry Point

Flattening a Package



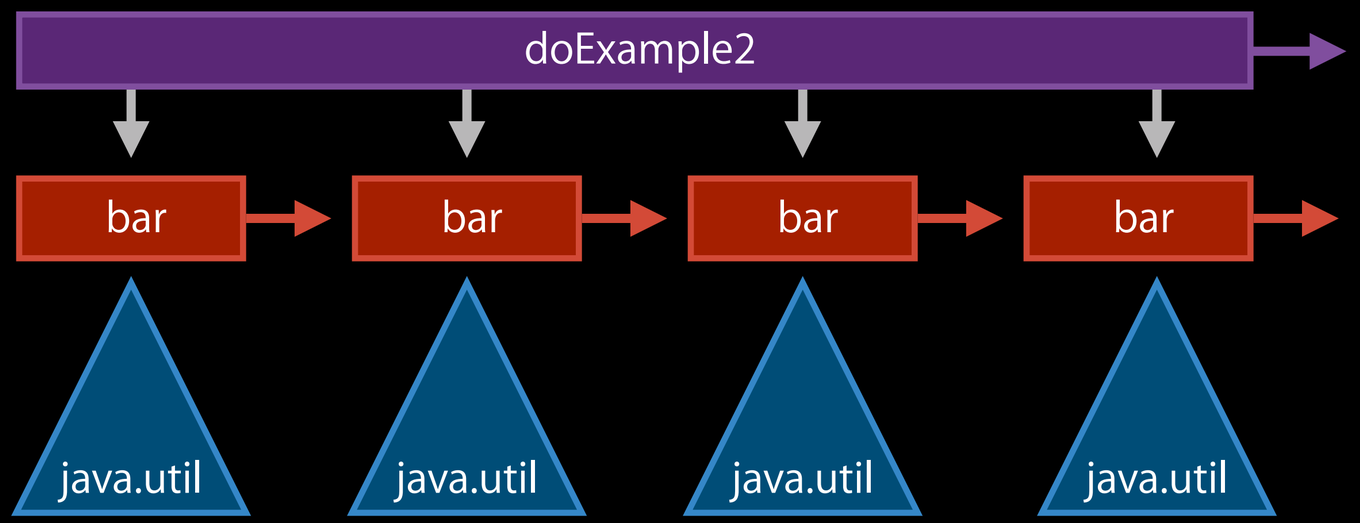
Focus on a Symbol

Focus on a method



Focus on a Symbol

Focus on a method



Hands-On Bouncy Text

Christy Warren

http://developer.apple.com/wwdc2005/

Shark For Java Part 2

Josh Outwater
Java Classes Engineer

http://developer.apple.com/wwdc2005/

Fractal Demo

- Generate fractals using a recursive algorithm
- Paint the resulting data as a height map to the screen
- Use Shark to determine where our “bottlenecks” are

Hands-On
Fractals

Josh Outwater

http://developer.apple.com/wwdc2005/

More Information

http://developer.apple.com/wwdc2005/

Documentation, Sample Code, and Other Resources

http://developer.apple.com/wwdc2005/

Related Sessions

Maximizing Java Performance	Lab Friday 10:30AM
Java and WebObjects Feedback Forum	Session 641 Friday 3:30PM

Q&A

Alan Samuel
bluker1@apple.com

http://developer.apple.com/wwdc2005/

WWDC2005

Worldwide Developers Conference